Fréchet mean and p-mean on the unit circle: characterization, decidability, and algorithm

Frédéric Cazals, Bernard Delmas, Timothée O'Donnell

Fréchet mean and p-mean on the unit circle: characterization, decidability, and algorithm

F. Cazals*and B. Delmas†and T. O'Donnell‡

February 19, 2020

Abstract

The center of mass of a point set lying on a manifold generalizes the celebrated Euclidean centroid, and is ubiquitous in statistical analysis in non Euclidean spaces. In this note, we give a complete characterization of the weighted p-mean of a finite set of angular values on S^1, based on a decomposition of S^1 such that the functional of interest has at most one local minimum per cell. This characterization is used to show that the problem is decidable for rational angular values –a consequence of Lindemann's theorem on the transcendence of π, and to develop an effective algorithm parameterized by exact predicates. A robust implementation of this algorithm based on multi-precision interval arithmetic is also presented. This implementation is effective for large values of n and p. Experiments on random sets of angles and protein dihedral angles consistently show that the Fréchet mean ($p = 2$) yields a variance reduction of $\sim 20\%$ with respect to the classically used circular mean.

Our derivations are of interest in two respects. First, efficient p-mean calculations are relevant to develop principal components analysis on the flat torus encoding angular spaces–a particularly important case to describe molecular conformations. Second, our two-stage strategy stresses the interest of combinatorial methods for p-means, also emphasizing the role of numerical issues.

The implementation is available in the Structural Bioinformatics Library (`http://sbl.inria.fr`).

Keywords: Frechét mean, p-mean, data centering, principal component analysis, circular statistics, decidability, robustness, multi-precision, angular spaces, molecular conformations

1 Introduction

1.1 Statistics on manifolds and p-means on S^1

Fréchet mean and generalizations. The celebrated center of mass of a point set P in a Euclidean space is the (a) point minimizing the sum of squared Euclidean to points in P. The center of mass plays a key role in data analysis at large, and in particular in principal components analysis since the data are centered prior to computing the covariance matrix and the principal directions. Generalizing these notions to non Euclidean spaces is an active area of research, as many mathematical objects in science and engineering turn out to live on Riemannian manifolds. Motivated by applications in structural biology (molecular conformations), robotics (robot conformations), and medicine (shape and relative positions of organs), early work focused on direct generalization of Euclidean notions. Analysis tailored to the unit circle and sphere were developed under the umbrella of directional statistics [AJ91, MT93, MJ09]. In a more abstract setting, generalizations of the center of mass in general metric spaces were first worked out – the so-called Fréchet mean [Fré48], followed by a generalization to distributions on such spaces – the so-called Karcher man

*Université Côte d'Azur, Inria, France
†INRAe, France
‡Université Côte d'Azur, Inria, France and INRAe, France

[GK73]. In differential geometry, recent developments focused on the existence, uniqueness and calculation of such objects [AM14, Pen18].

In fact, previous works span two complementary directions. On the one hand, efforts have focused on mathematical properties of spaces generalizing affine spaces, so as to provide statistical summaries of ensembles in terms of geometric objects of small dimension. On the other hand, algorithmic developments have been performed to compute such objects. The case of the unit circle S^1 provides the simplest compact non Euclidean manifold to be analyzed. Despite its simplicity, this case turns out to be of high interest since S^1 encodes angles, a particularly important case e.g. to describe molecular conformations. In the sequel, we focus on p-means defined on the unit circle S^1, for $p \geq 1$.

Consider n angles $\Theta_0 = \{\theta_i\}_{i=1,\dots,n}$. Practically, in processing real data, we shall consider that each angle is given as a rational number. Consider the embedding of an angle onto the unit circle, that is $X(\theta) = (\cos\theta, \sin\theta)^\intercal$. The geodesic distance between two points $X(\theta)$ and $X(\theta_i)$ on S^1, denoted $d(\cdot,\cdot)$, satisfies

$$d(X(\theta), X(\theta_i)) = 2 \arcsin \frac{\|X(\theta) - X(\theta_i\|}{2}. \tag{1}$$

This distance is also directly expressed using the angles:

$$d(X(\theta), X(\theta_i)) = \min(\mid \theta - \theta_i \mid, 2\pi - \mid \theta - \theta_i \mid). \tag{2}$$

Consider a set of positive weights $\{w_i\}_{i=1,\dots,n}$. For an integer $p \geq 1$, consider the function involving the weighted distances to all points, i.e.

$$F_p(\theta) = \sum_{i=1,\dots,n} w_i f_i(\theta), \text{ with } f_i(\theta) = d^p(X(\theta), X(\theta_i)). \tag{3}$$

We denote its minimum

$$\theta^* = \arg \min_{\theta \in [0, 2\pi)} F_p(\theta). \tag{4}$$

For units weights, the value θ^* obtained for $p = 2$ corresponds to the Fréchet mean.

The previous expression can be seen as a distance to a point mass probability distribution on S^1. For a general probability distribution on S^1, necessary and sufficient conditions for the existence of a Fréchet mean have been worked out [Cha13]. In the same paper, the author proposes a quadratic algorithm–regardless of numerical issues–to compute the Fréchet mean for the particular case of a point mass probability distribution. In a more general setting, a stochastic algorithm finding p-means wrt a general measure on the circle has also been proposed [AM16].

Remark 1. *In the following the analysis presented in sections 2 and 3 does not depend on the weights in Eq. 3 – assuming these are rational numbers. To make notations lighter, we therefore omit them in the sequel of the paper. However, our implementation does incorporate them.*

Robustness and numerical issues. From the optimization standpoint, computing p-means on S^1 turns out to be a non convex problem, which can be decomposed into piecewise convex problems. The simplest model of a computer to solve such (geometric) problems is the real RAM computer model, which assumes that exact operations on real numbers are available at constant time per operation [PS85]. This model is unrealistic due to rounding operations inherent to floating point representations [MBdD$^+$18]. In particular, erroneous evaluations of decisions due to rounding errors may trigger wrong decisions, and the algorithm may loop, crash, or terminate with an wrong answer, a situation occurring even for the simplest 2D geometric calculations [KMP$^+$08].

Robust geometric algorithms, which deliver what they are designed for, can be developed using the Exact Geometric Computation (EGC) paradigm [YD95], which is central in the Computational Geometry Algorithms Library (CGAL) [cga]. The EGC relies on so-called *exact predicates* and *constructions*. A predicate is a function whose output belongs to a finite set, while a construction exhibits a new geometric object from the input data. For example, the predicate `Sign`(x) returns the sign $\{negative, null, positive\}$ of

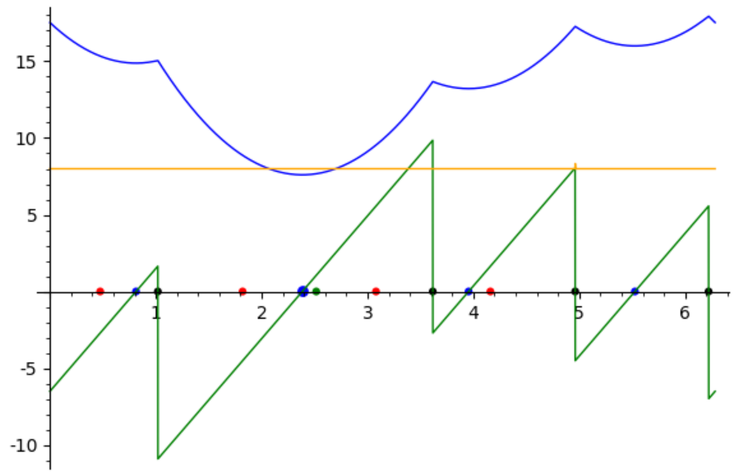


Figure 1: **Fréchet mean of four points on S^1 (Functions)** blue: function F_2; green: derivative $F_2^{'}$; orange: second derivative $F_2^{''}$ **(Points)** red bullets: data points; black bullets: antipodal points; blue bullets: local minima of the function; large blue bullet: Fréchet mean θ^*; green bullet: circular mean Eq. 23.

the arithmetic expression x. As we shall see, designing robust predicates for p-means on S^1 turns out to be connected to transcendental number theory since expressions involving π are dealt with. In particular, one needs to evaluate the sign of such expressions, which raises decidability issues. May be not so surprisingly, we note in passing that very few geometric problems involving transcendental numbers have been shown to be decidable–see e.g. [CCK$^+$06]. Fortunately, we shall see that our problem is so.

1.2 Contributions

This paper makes three contributions regarding p-means of a finite point set. First, we show that the function F_p is determined by a very simple combinatorial structure, namely a partition of S^1 into circle arcs. Second, we give an explicit expression for F_p, deduce that the problem is decidable, and present an algorithm computing p-means. Third, we present an effective and robust implementation, based on multi-precision interval arithmetic.

2 p-mean of a finite point set on S^1: characterization

2.1 Notations

In the following, angles are in $[0, 2\pi)$. We first define:

Definition. 1. *For each angle $\theta_i \in [0, \pi)$, we define $\theta_i^+ = \theta_i + \pi$. The set of all such angles is denoted $\Theta^+ = \{\theta_i^+\}$ For each angle $\theta_i \in [\pi, 2\pi)$, we define $\theta_-^+ = \theta_i - \pi$. The set of all such angles is denoted $\Theta^- = \{\theta_i^-\}$. The* antipodal set *of Θ_0 is the set of angles $\Theta^\pm = \Theta^+ \cup \Theta^-$.*

Altogether, these angles yield the larger set

$$\Theta = \Theta_0 \cup \Theta^\pm. \tag{5}$$

Note that $|\Theta| = 2n$ since each angle in Θ_0 also contributes one value in Θ^+ or Θ^+. Angles in Θ are generically denoted α_i or α_j. Note however that when referring to an angle in the continuous interval $[0, 2\pi)$, θ is used.

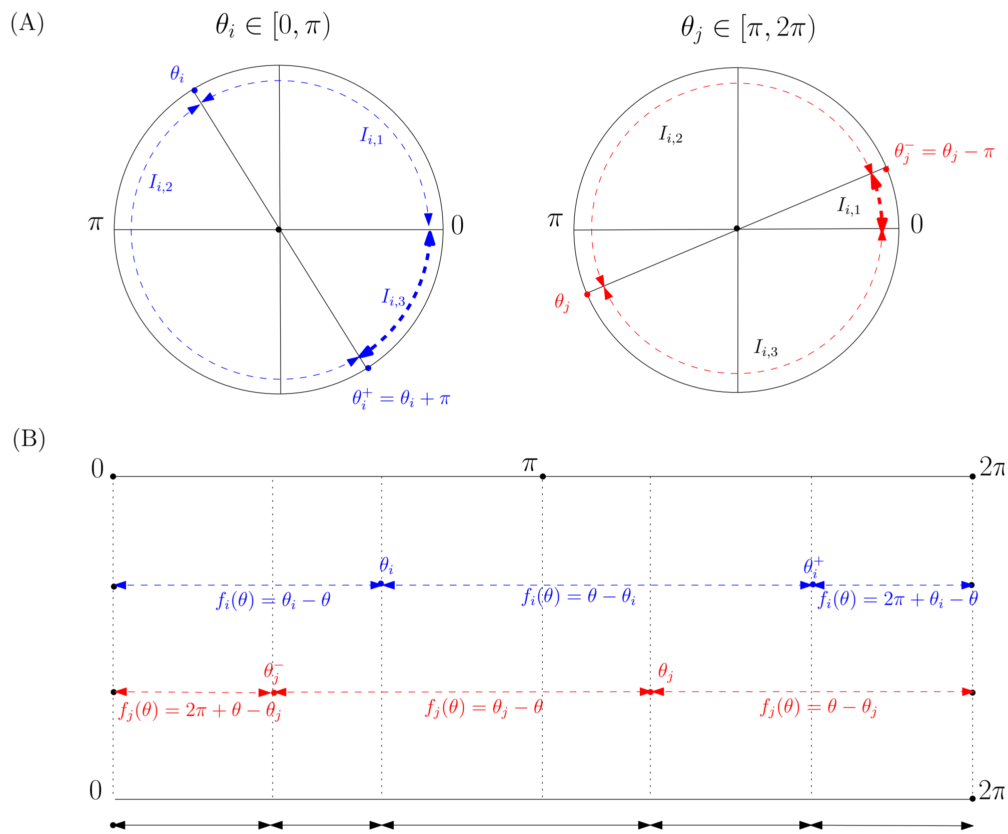


Figure 2: **The partition of S^1 into circle arcs underlying the function F_p (A)** The three elementary intervals defined by angles in $[0, \pi)$ and $[\pi, 2\pi)$ respectively. Bold circle arcs indicate that f_i has a transcendental expression i.e. involves π. **(B)** Intersections of elementary intervals in $\mathcal{I}$ yield a partition of $[0, 2\pi)$, and associated piecewise functions.

To each angle θ_i, we associate three so-called *elementary intervals* (Fig. 2(A)):

- $\theta_i \in [0, \pi) : I_{i,1} = (0, \theta_i), I_{i,2} = (\theta_i, \theta_i^+), I_{i,3} = (\theta_i^+, 2\pi)$.
- $\theta_i \in [\pi, 2\pi) : I_{i,1} = (0, \theta_i^-), I_{i,2} = (\theta_i^-, \theta_i), I_{i,3} = (\theta_i, 2\pi)$.

2.2 Partition of S^1

We also consider the partition of $[0, 2\pi)$ induced by the intersection of the $3n$ intervals $\{I_{i,1}, I_{i,2}, I_{i,3}\}$ (Fig. 2(B)). More specifically, we choose one interval (out of three) for each function f_i, and intersect them all:

Definition. 2. *The elementary intervals $I_{i,j}$ define a partition of S^1 based on the following intervals:*

$$\mathcal{I} = \{ \bigcap_{i=1,\dots,n} (I_{i,1} \vee I_{i,2} \vee I_{i,3}) \text{ with } \bigcap_{i=1,\dots,n} I_{i,j} \neq \emptyset. \}. \tag{6}$$

In the sequel, open intervals from $\mathcal{I}$ are denoted (α_j, α_{j+1}).

Remark 2. *From the previous definition, it appears that the intervals in $\mathcal{I}$ may be ascribed to nine types since the left endpoint is an angle θ_i or an antipodal angle θ_i^+ or θ_i^-, and likewise for the right endpoint.*

2.3 Piecewise expression for F_p

Function F_p. We use the previous intervals to unveil the piecewise structure of F_p. We define the following piecewise functions (Fig 2(B)):

$$\theta_i \in [0, \pi) : f_i(\theta) = \begin{cases} (\theta_i - \theta)^p, \text{ for } \theta \in I_{i,1}, \\ (\theta - \theta_i)^p, \text{ for } \theta \in I_{i,2}, \\ (2\pi + \theta_i - \theta)^p, \text{ for } \theta \in I_{i,3}. \end{cases} \tag{7}$$

$$\theta_i \in [\pi, 2\pi) : f_i(\theta) = \begin{cases} (2\pi + \theta - \theta_i)^p, \text{ for } \theta \in I_{i,1}, \\ (\theta_i - \theta)^p, \text{ for } \theta \in I_{i,2}, \\ (\theta - \theta_i)^p, \text{ for } \theta \in I_{i,3}. \end{cases} \tag{8}$$

Remark 3. *Let θ_{max} be the antipodal value of the largest $\theta_i \in \Theta_0$ larger than π, and θ_{min} the antipode of the smallest $\theta_i \in \Theta_0$ smaller than π. Function F_p is transcendental in $[0, \theta_{max})$ and $(\theta_{min}, 2\pi]$ – its expression involves π. Also, function F_p is algebraic on $(\theta_{max}, \theta_{min})$. See Fig. 2.*

Derivative of F_p. To study local minima of F_p, we focus on derivatives of F_p up to the second order. To this end, we study the derivatives of each function f_i. Since each f_i is obviously smooth on open intervals, and computing the derivatives of the terms in Eqs. 7 and 8 yields:

$$\theta_i \in [0, \pi) : f_i'(\theta) = \begin{cases} -p(\theta_i - \theta)^{p-1}, \text{ for } \theta \in I_{i,1}, \\ p(\theta - \theta_i)^{p-1}, \text{ for } \theta \in I_{i,2}, \\ -p(2\pi + \theta_i - \theta)^{p-1}, \text{ for } \theta \in I_{i,3}, \end{cases} \tag{9}$$

$$\theta_i \in [\pi, 2\pi) : f_i'(\theta) = \begin{cases} p(2\pi + \theta - \theta_i)^{p-1}, \text{ for } \theta \in I_{i,1} \\ -p(\theta_i - \theta)^{p-1}, \text{ for } \theta \in I_{i,2} \\ p(\theta - \theta_i)^{p-1}, \text{ for } \theta \in I_{i,3}. \end{cases} \tag{10}$$

To study possible discontinuities at points in $\alpha_j \in \Theta$, we define:

$$\Delta f'_{i|\theta} = \lim_{\theta \searrow \alpha_j} f_i'(\theta) - \lim_{\theta \nearrow \alpha_j} f_i'(\theta) \tag{11}$$

We now apply these formulae, distinguishing two cases, namely $p = 1$ and $p > 1$.

Properties of F_p: case $p = 1$.

Lemma. 1. *For $p = 1$, function f_i and its derivative satisfy:*

- *Function f_i is continuous on S^1.*
- *Function f_i' continuous on S^1 except at θ_i and its antipodal value; at these points, $\Delta f'_{i|\theta}$ is equal to 2 or -2.*

Proof. For the continuity, one has

$$\theta_i \in [0, \pi) : f_i(\theta_i) = 0, \;\; f_i(\theta_i^+) = \pi, f_i(0) = \theta_i \tag{12}$$

$$\theta_i \in [\pi, 2\pi) : f_i(\theta_i^-) = \pi, f_i(\theta_i) = 0, \;\; f_i(0) = 2\pi - \theta_i. \tag{13}$$

For changes of the derivative at points and antipodal points, we get from Eqs. 9 and 10:

$$\begin{cases} \alpha_j = \theta_i \in [0,\pi) : \begin{cases} \Delta f'_{i|\theta_i} = 2 \\ \Delta f'_{i|\theta_i^+} = -2 \\ \Delta f'_{i|0} = 0 \end{cases} \\ \alpha_j = \theta_i \in [\pi, 2\pi) : \begin{cases} \Delta f'_{i|\theta_i^-} = -2 \\ \Delta f'_{i|\theta_i} = 2 \\ \Delta f'_{i|0} = 0 \end{cases} \end{cases} \tag{14}$$

□

Using the previous properties, the global behavior of F_1 is as follows:

Lemma. 2. *Local minima of the function F_1 are either isolated points or plateaus.*

Proof. On any open interval from $\mathcal{I}$, function F_1 is linear, which reflects the fact that when θ increases, we approach or move away with respect to any θ_i. Given the possible options for the derivative drops – Eq. 14, at any angle from Θ, we face three options for the derivative: its sign does not change in which case the function keeps increasing or decreasing; its sign changes – in which case the function admits a local extremum; or it become null, in which case the function admits a local minimum which is a plateau. This latter case occurs when n is even, for an interval admitting the same number of of points from Θ_0 on each side. □

Properties of F_p: case $p > 1$.

Lemma. 3. *For $p > 1$, function f_i and its derivatives satisfy:*

- *Function f_i is continuous on S^1.*
- *The derivative f'_i is continuous on S^1 except at the antipodal value of θ_i, where $\Delta f'_{i|antipode(\theta_i)} = -2p\ \pi^{p-1}$.*
- *The second order derivative f''_i is non negative on S^1.*

Proof. For the continuity, one has

$$\theta_i \in [0,\pi) : f_i(\theta_i) = 0, \quad f_i(\theta_i^+) = \pi^p, f_i(0) = \theta_i^p \tag{15}$$

$$\theta_i \in [\pi, 2\pi) : f_i(\theta_i^-) = \pi^p, f_i(\theta_i) = 0, \quad f_i(0) = (2\pi - \theta_i)^p. \tag{16}$$

For changes of the derivative at points and antipodal points, we get from Eqs. 9 and 10:

$$\begin{cases} \alpha_j = \theta_i \in [0,\pi) : \begin{cases} \Delta f'_{i|\theta_i} = 0 \\ \Delta f'_{i|\theta_i^+} = -2p\ \pi^{p-1} \\ \Delta f'_{i|0} = 0. \end{cases} \\ \alpha_j = \theta_i \in [\pi, 2\pi) : \begin{cases} \Delta f'_{i|\theta_i^-} = -2p\ \pi^{p-1} \\ \Delta f'_{i|\theta_i} = 0 \\ \Delta f'_{i|0} = 0. \end{cases} \end{cases} \tag{17}$$

Finally, the positivity of $f_i^{''}$ stems from the second derivative of terms in Eqs. 9 and 10. □

The previous lemma tells us that $F_p^{'}$ incurs drops at antipodal points, and then keeps increasing again on the interval starting at that point. Finding local minima of F_p therefore requires finding those intervals from $\mathcal{I}$ where $F_p^{'}$ vanishes, which happens at most once. We conclude with:

Lemma. 4. *For $p > 1$, function F_p has at most one local minimum on each interval in $\mathcal{I}$.*

3 Algorithm

The observations above are not sufficient to obtain an efficient algorithm: since there are $2n$ intervals and since the function has linear complexity on each of them, a linear number of function evaluations has quadratic complexity. We get around this difficulty by maintaining the expression of the function at angles in Θ.

3.1 Analytical expressions and nullity of $F_p^{'}$

Function F_p and its derivative. We first derive a compact, analytical expression of F_p and $F_p^{'}$. Following Eqs. 7 and 8, the expressions of $f_i(\theta)$ and $f_i'(\theta)$ can be written as

$$f_i'(\theta) = k_i \times (a_i + \varepsilon_i \theta)^{p-1}, \text{ with } k_i \in \{-p, p\}, a_i \in \{-\theta_i, 2\pi - \theta_i, \theta_i, 2\pi + \theta_i\}, \varepsilon_i \in \{-1, +1\}. \quad (18)$$

On open intervals (α_j, α_{j+1}), the function reads as the following polynomial

$$F_p(\theta) = \sum_{i=1}^{n} (a_i + \varepsilon_i \theta)^p = \sum_{j=0}^{p} b_j \theta^j, \text{ with } b_j = \sum_{i=1}^{n} \binom{p}{j} a_i^{p-j} \varepsilon_i^j. \quad (19)$$

Similarly, the derivative $F_p^{'}(\theta)$ reads as a degree $p-1$ polynomial:

$$F_p^{'}(\theta) = \sum_{i=1}^{n} k_i (a_i + \varepsilon_i \theta)^{p-1} = \sum_{j=0}^{p-1} c_j \theta^j, \text{ with } c_j = \sum_{i=1}^{n} k_i \binom{p-1}{j} a_i^{p-1-j} \varepsilon_i^j. \quad (20)$$

Nullity of $F_p^{'}$: algebraic versus transcendental expressions. The previous equations call for two important comments. First, from the combinatorial complexity standpoint, if the coefficients of the polynomials are known, evaluating F_p and $F_p^{'}$ has cost $O(p)$. Second, from the numerical standpoint, locating local minima of F_p requires finding intervals from $\mathcal{I}$ on which $F_p^{'}$ vanishes. Identifying such intervals is key to the robustness of our algorithm. Practically, since an interval is defined by two consecutive values in the set Θ, we need to check that the sign of $F_p^{'}$ differs at these endpoints. The cornerstone is therefore to decide the sign of $F_p^{'}$ at angles in Θ (input angles or their antipodes), and the following is a simple consequence of Lindemann's theorem on the transcendence of π:

Lemma. 5. *If the angular values $\theta_i \in \Theta_0$ are rational numbers, checking whether $F_p^{'}(\alpha_i) \neq 0$ for any $\alpha_i \in \Theta$ is decidable. Moreover, when $F_p^{'}$ has a transcendental expression and α_i is rational, $F_p^{'} \neq 0$.*

Proof. We first consider the case $\alpha_i \in \Theta_0$, and distinguish the two types of intervals – see Rmk 3.
First, consider an interval where F_p has an algebraic expression. We face a purely algebraic problem, and deciding whether $F_p^{'}(\alpha_i) \neq 0$ can be done using classical bounds, e.g. Mahler bounds [LPY05, YYD+10].
Second, consider an interval where F_p has a transcendental expression. Then, $F_p^{'}(\alpha_i)$ can be rewritten as a polynomial of degree $p-1$ in π. Lindemann's theorem on the transcendence of π implies that $F_p^{'}(\alpha_i) \neq 0$.

Consider now the case where $\alpha_i \in \Theta^{\pm}$, that is $\alpha_i = \alpha_j \pm \pi$. Each individual term ${f_i}'(\alpha_i)$ also has the form $(c_i\pi + q_i)^{p-1}$, with $c_i \in \mathbb{N}$ and $q_i \in \mathbb{Q}$, so that the latter case also applies. □

3.2 Algorithm: case $p = 1$

We sort the angles in Θ and initialize $F_p^{'}(\theta)$ at the value it has before the first element of Θ. When crossing a sample point $F_p^{'}$ increases by 2 (Lemma 1). When crossing an antipodal point $F_p^{'}$ decreases by two. The nature of local minima is given by lemma 2. If the number of sample points is odd whenever crossing a sample point if the derivative goes from -1 to 1 this point is a minimum. If the number of points is even whenever there are two consecutive sample points if the derivative goes from -2 to 0 to 2, then, the circle arc between these two points is an extended local minimum – a plateau.

Assume the coefficients of F_p have been stored in a vector B of size $p+1$. As noticed above, evaluating F_p at a given angle has cost $O(p)$. The overall algorithms therefore has complexity $O(n \log n + np)$.

3.3 Algorithm: case $p > 1$

Following the previous analysis, finding the p-mean(s) involves the following five steps (see Algorithm 1):

- (Step 0) Construct the set Θ;
- (Step 1) Sort angles in Θ;
- For all intervals in $\mathcal{I}$:
  - (Step 2) Identify the intervals where $F_p^{'}$ vanishes;
  - (Step 3) Compute the unique root of $F_p^{'}$;
  - (Step 4) Evaluate F_p at a local minimum;
  - (Step 5) Maintain the polynomials F_p and $F_p^{'}$.

In the sequel, we analyze the complexity of these steps.

(Step 0) Construct the set Θ. This step has linear complexity.

(Step 1) Sort angles in Θ. This step has complexity $O(n \log n)$.

(Step 2) Identify the intervals where $F_p^{'}$ vanishes. By lemmas 3 and 4, there is at most one local minimum per interval, which requires checking the signs of $F_p^{'}$ to the right and left bounds of an interval (α_j, α_{j+1}). Using the functional forms encoded in vector C, computing these derivatives has the same complexity as the previous step. However, this step calls for two important comments:

- For $\alpha_i \in \Theta$, checking whether $F_p^{'}(\alpha_i) \neq 0$ is decidable – lemma 5. However, the arithmetic nature of the number α_i must be taken into account, as rational numbers (input angles) and transcendental numbers (antipodal points) must be dealt with using different arithmetic techniques. See below.
- Not all intervals (α_j, α_{j+1}) can provide a root. Indeed, once $F_p^{'}(\alpha_i) > 0$, since the individual second order derivatives are positive, $F_p^{'}$ cannot vanish until one crosses one $\alpha_j \in \Theta^{\pm}$. As we shall see, this observation is easily accommodated in algorithm 1.

In the sequel, we denote $\mathrm{SD}(p-1)$ the cost of deciding the sign (negative, zero, positive) of $F_p^{'}(\theta)$, for $\theta \in \Theta$.

(Step 3) Compute the unique root of $F_p^{'}$. Since $F_p^{'}$ is piecewise polynomial, finding its real root has constant time complexity for $p \leq 5$. Otherwise, a numerical method can be used [KRS16].

In the sequel, we denote $\mathrm{RF}(p-1)$ the cost of isolating the real root of a degree $p-1$ polynomial.

(Step 4) Evaluate F_p at a local minimum. Once the angle θ_m corresponding to a local minimum has been computed, we evaluate $F_p(\theta_m)$ using Eq. 19. This evaluation has $O(p)$ complexity since the coefficients of the polynomial are known.

(Step 5) Maintain the polynomials F_p and $F_p^{'}$. Following Eqs. 19 and 20, the function and its derivative only change when crossing an angle from Θ. At such an angle, updating the vectors B and C has complexity $O(p)$. Overall, this step therefore has complexity $O(np)$.

We summarize with the following output-sensitive complexity:

Theorem. 1. *Computing the p-mean has $O(n \log n + np + nSD(p-1) + kRF(p-1) + kp)$ complexity, with k the number of local minima of F_p.*

Remark 4. *Note that a naive (i.e. without the maintenance of the coefficients of F_p) calculation of F_p at local minima has quadratic combinatorial complexity in the case of a linear number of local minima.*

3.4 Generic implementation

In the following, we present an implementation of our algorithm based on predicates, i.e. functions deciding branching points.

Pseudo-code, predicates and constructions Our algorithm (Algo. 1) takes as input a list of angular values (in degrees or radians) and the value of p. Following Rmk 1, an optional file containing the weights may be passed. If $p > 5$, we take for granted an algorithm computing the root of $F_p^{'}$ on an interval. As a default, we resort to a bisection method which divides the interval into two, checks which side contains the unique root of $F_p^{'}$, and iterates until the width of the interval is less than some user specified value τ (Algo. 3). The interval returned is called the *root isolation interval*. Our algorithm was implemented in generic C++ in the Structural Bioinformatics Library [CD17], as a template class whose main parameter is a geometric kernel providing the required predicates and constructions. We now discuss these–see Sec. 3.5 for their robust implementation.

Predicates. The algorithm involves two predicates:

- `Sign`$(F_p^{'}(\theta))$. Predicate used to determine the sign of the $F_p^{'}(\theta)$ with $\theta \in [0, 2\pi)$ (Algo. 3).
- `Interval_too_wide`(θ_l, θ_r). Predicate used to determine whether the root isolation interval has width less than τ (Algo. 3). It is true if $\theta_r - \theta_l > \tau$, and false otherwise.

Constructions.

- **Updating constants**. Updating the coefficients in B and C is necessary at each $\alpha_i \in \Theta$: for $F_p(\theta)$ (resp. $F_p^{'}(\theta)$), we subtract the contribution of $f_i(\theta)$ (resp. $f_i^{'}(\theta)$) before α_i, and add that of $f_i(\theta)$ (resp. $f_i^{'}(\theta)$) after α_i.
- **Find_root**. Computing the root of $F_p^{'}$ on an interval (α_j, α_{j+1}) is handled in two ways. If $p \leq 5$, radical based formulae can be implemented. If $p > 5$ a root finding algorithm must be implemented. Our implementation resorts to a bisection method for $p > 3$(Algo. 3).

Remark 5. *A kernel based on floating point number types, the* `double` *type in our case, is easily assembled, see* `SBL::GT::Inexact_predicates_kernel_for_frechet_mean` *in Sec. 5. As noticed earlier, it comes with no guarantee. In particular, the algorithm may terminate with an erroneous result if selected predicates are falsely evaluated.*

Algorithm 1 p-mean calculation: generic algorithm for $p > 1$ in the real RAM model

Θ: $vector[1, 2n]$ containing all the angles
B: $vector[1, p+1]$ to store the coefficients of the polynomial $F_p(\theta)$ Eq. 19
C: $vector[1, p]$ to store the coefficients of the polynomial $F_p^{'}(\theta)$ Eq. 20
θ^* // Angle corresponding to the global minimum of F_p
Root_remains = true // flag indicating whether a root must be sought on (α_j, α_{j+1})

// Initialization
Compute $\Theta^{\pm}$ and form sorted Θ
α_0: first angle in Θ
Store the coefficients of F_p into the vector B for the interval $(0, \alpha_0)$
Store the coefficients of $F_p^{'}$ into vector C for the interval $(0, \alpha_0)$
Compute $l \leftarrow F_p^{'}(\theta)$ for $\theta \to 0^+$ using Eq. 20 and vector C
Update_root(`Sign`(l))//Updates Root_remains see Algo. 2
if `Sign`(l) is null **then**
    Compute $F_p(0)$ using vector B and Eq. 19, and possibly update θ^*.

// For each angle: handle interval ending, update coefficients in B and C, handle interval starting
for all α_i **in** Θ **do**
    if Root_remains **then**
        Compute $r \leftarrow F_p^{'}(\theta)$ for $\theta \to \alpha_i^-$ using Eq. 20 and vector C
        Update_root(`Sign`(r))//Updates Root_remains see Algo. 2
        if `Sign`(r) is positive **then**
            $\theta_c \leftarrow$ **Find_root**(α_{i-1}, α_i)
            Compute $F_p(\theta_c)$ using vector B and Eq. 19, and possibly update θ^*.
        else if `Sign`(r) is null **then**
            Compute $F_p(\alpha_i)$ using vector B and Eq. 19, and possibly update θ^*.
    Update the coefficients of F_p stored in vector B upon crossing α_i
    Update the coefficients of $F_p^{'}$ stored in vector C upon crossing α_i
    if $\alpha_i \in \Theta^{\pm}$ **then**
        Compute $l \leftarrow F_p^{'}(\theta)$ for $\theta \to \alpha_i^+$ using Eq. 20 and vector C
        Update_root(`Sign`(l))//Updates Root_remains see Algo. 2
        if `Sign`(l) is null **then**
            Compute $F_p(\alpha_i)$ using vector B and Eq. 19, and possibly update θ^*.

// Process the interval ending at 2π
Compute $r \leftarrow F_p^{'}(\theta)$ for $\theta \to 2\pi^-$ using Eq. 20 and vector C
if Root_remains **then**
    if `Sign`(r) is positive **then**
        $\theta_c \leftarrow$ **Find_root**$(\theta_{2n}, 2\pi)$
        Compute $F_p(\theta_c)$ using vector B and Eq. 19, and possibly update θ^*
    else if `Sign`(r) is null **then**
        Compute $F_p(2\pi)$ using vector B and Eq. 19, and possibly update θ^*.

Algorithm 2 Update_root($Sign$): Updates the Root_remains buffer in main algorithm(Algo. 1)

$Sign$ ∈{positive,negative,null} Sign of the derivative used to update the presence of roots on (α_j, α_{j+1})
Root_remains = true // flag indicating whether a root must be sought on (α_j, α_{j+1})
if $Sign$ is negative **then**
    Root_remains = true
else if $Sign$ is positive **then**
    Root_remains = false
else if $Sign$ is null **then**
    Root_remains = false

Algorithm 3 Find_root(α_{i-1}, α_i): generic algorithm for $p > 5$

α_{i-1}, α_i: the left and right endpoints of the initial interval
τ: Threshold to stop binary search if interval is small enough
c: Center of interval g
$\theta_l = \alpha_{i-1}, \theta_r = \alpha_i$ // Interval being bisected
while `Interval_too_wide`(θ_l, θ_r) **do**
    Compute $c = \theta_l + (\theta_r - \theta_l)/2$
    Compute $S =$ `Sign`$(F_p^{'}(c))$
    if S is positive **then**
        $\theta_r = c$
    else if S is negative **then**
        $\theta_l = c$
    else if S is null **then**
        $\theta_r = c$
        $\theta_l = c$
Compute $\theta_c = \theta_l + (\theta_r - \theta_l)/2$

3.5 Robust implementation based on exact predicates

Number types for lazy evaluations. Following the Exact Geometric Computation exact predicates are gathered in a *kernel*. We circumvent rounding errors using interval number types which are certified to contain the exact value of interest. That is, an expression x is represented by the interval $[\underline{x}, \overline{x}] \ni x$. The bounds of these intervals may have a fixed precision, which corresponds to the CGAL::Interval_nt number type [cga]. Or the bounds may be multiprecision, e.g. `Gmpfr` from `Mpfr`; [FHL+07], which corresponds to the CGAL::Gmpfi type [cga]. We now explain how these types are used to code exact predicates.

The `Sign` predicate. We distinguish the algebraic and transcendental cases, performing multiprecision calculations only if needed (Fig. 3).

•**Transcendental case: multiprecision interval arithmetic.** When F_p is transcendental and α_i rational, $F_p^{'}(\alpha_i)$ is positive or negative (lemma 5). Another case where $F_p^{'}(\alpha_i) \neq 0$ is when $\alpha_i \in \Theta^{\pm}$. In our implementation this situation is faced in two cases. First, in the main algorithm (Algo. 1), `Sign`(l) or `Sign`(r): l and r are transcendental if $\alpha_i \in \Theta^{\pm}$. Second, in the root finding algorithm(Algo. 3), `Sign`($F_p^{'}(c)$): c is transcendental if α_{i-1} or $\alpha_i \in \Theta^{\pm}$. In both cases, we proceed in a lazy way: first, we try to conclude using CGAL::Interval_nt; if this interval contains zero, we switch to CGAL::Gmpfi (Fig. 3), refine the interval bounds, and conclude. Refining the interval consists of iteratively doubling the number of bits used to describe all numbers–including π, until a conclusion can be reached.

•**Algebraic case: zero separation bounds.** When F_p has a rational expression and α_i is rational, `Sign`($F_p^{'}(\alpha_i)$) may be zero. Fig. 4 presents a simple case with three angles. In this case, an input angle may also corresponds to a local minimum of F_p. To decide whether $F_p^{'}(\alpha_i) = 0$, we resort to zero separation bounds and multiprecision interval arithmetic.

Let us consider $F_p^{'}(\alpha_i)$ as an arithmetic expression E, using a number of authorized operations($\pm, \times, /$ in our case). A separation bound is a function *sep* such that the value ξ of expression E is lower bounded by *sep*(E) in the following manner:

$$If\ \xi \neq 0 \text{ then } \mathrm{sep}(E) \leq |\xi| \tag{21}$$

Considering $\tilde{\xi}$ an approximation of ξ and Δ an upper bounded error $|\tilde{\xi} - \xi|$.

$$If\ |\tilde{\xi}| + \Delta < \mathrm{sep}(E) \text{ then } \xi = 0. \tag{22}$$

Practically, we proceed in a lazy way, in two steps (Fig. 3). First, using CGAL::Interval_nt with double precision, we check whether we can conclude on $F_p^{'}(\alpha_i) \neq 0$. If not–the interval contains zero, we use CORE::ExprT[KLPY99] to determine the zero separation bound and decide if $F_p^{'}(\alpha_i) = 0$. If not, we finally determine the sign.

Predicate `Interval_too_wide`(θ_l, θ_r). Returns true when $\underline{\theta_r} - \overline{\theta_l} > \tau$, false if $\overline{\theta_r} - \underline{\theta_l} \leq \tau$. Similarly to the sign predicate, we distinguish the transcendental and algebraic cases to check whether $\theta_l - \theta_r - \tau = 0$. Supposing τ and Θ_0 are rational $\theta_l - \theta_r - \tau$ is transcendental if the initial α_{i-1} or $\alpha_i \in \Theta^{\pm}$. If transcendental the interval is refined in the same way as the transcendental case of the `Sign` predicate. Otherwise the expression is algebraic and the precision is raised until an exact computation can be performed.

4 Experiments

4.1 Overview

Our experiments target three properties, namely: (i) robustness to ascertain how often numerical refinement is triggered, (ii), comparison of the Fréchet mean against the classical circular mean, and (iii) computational complexity.

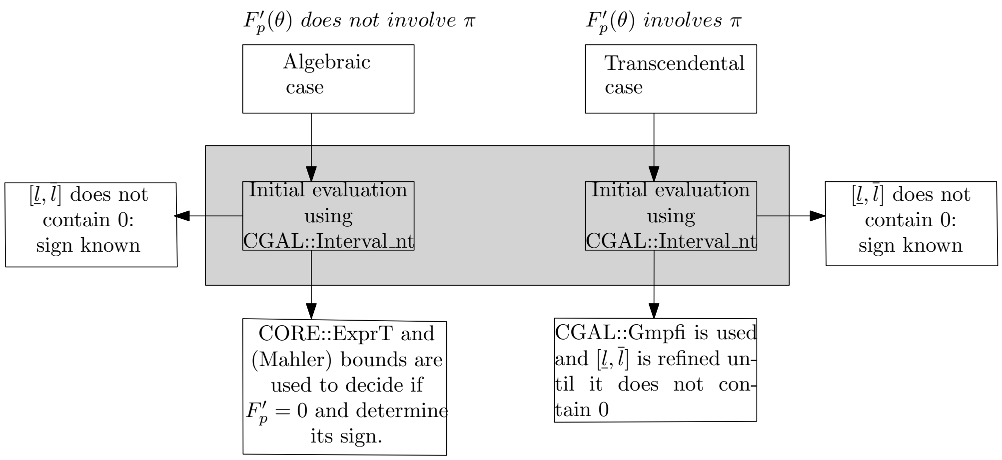
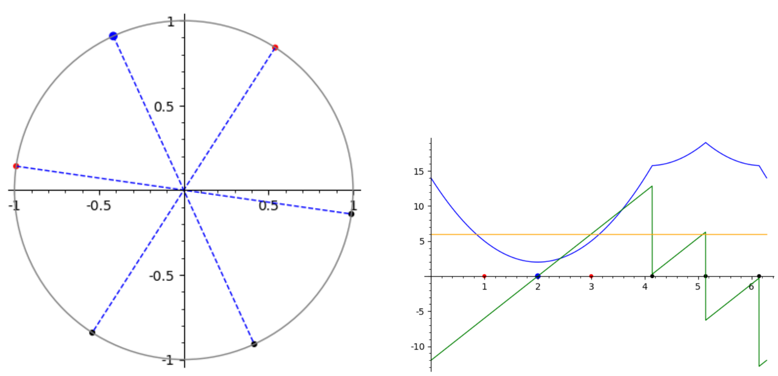


Figure 3: **Number types used in the `Sign` predicate.** Note that CGAL::Interval_nt is used in the algebraic and transcendental cases, while the remaining number types are only used if required.



$n = 3, p = 2$ $n = 3, p = 2$

Figure 4: **An interval where F_p has an algebraic expression and $F_p^{'}(\theta) = 0$.** Illustration of $F_p, F_p^{'}, F_p^{''}$ for $p = 2$ and three angles $\Theta_0 = \{\theta_1 = 1, \theta_2 = 2, \theta_3 = 3\}$. Color conventions as in Fig. 1. In this case, $F_2'(\theta_2) = 0$, which must be numerically ascertained to ensure the correctness of the algorithm.

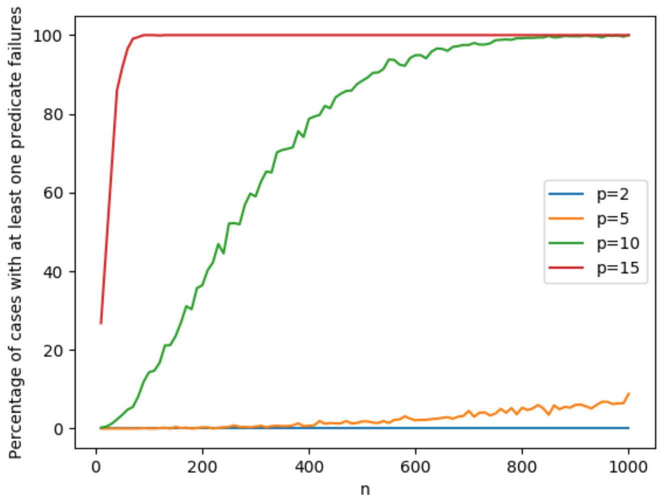


Figure 5: **Fraction of program runs for which at least one predicate execution triggers refinement, as a function of n and p.** The number of repeats for each value of n is 1000.

Practically, two sets of angles are used. First, randomly generated angles. Second, so-called dihedral angles in proteins. In brief, recall that the covalent structure of a protein is the graph whose nodes are the atoms and whose edges correspond to covalent bonds. Four any consecutive atoms define a dihedral angle – the angle between the planes defined by the first three and the last three atoms respectively. These angles are known to be dependent, and correlations between them are key to reduce the dimensionality of the conformation space of proteins [MT93, JS01, TWS$^+$10]. Using the Protein Data Bank, we retained 27093 PDB files with a resolution of 3 angstroms or better. For all polypeptide chains in these files, we computed all dihedral angles of all standard (20) amino-acids. This results in 240 classes of dihedral angles, containing from 50,227 to 439,793 observations.

4.2 Robustness

Using our robust interval-based implementation, we count the fraction of cases for which at least one predicate triggers refinement during an execution. We use sets of $n \in [10, 1000]$ angles generated uniformly at random in $[0, 2\pi)$, and perform 1000 repeats for each value of n (Fig. 5). For large values of p, whenever $n > 1000$, all executions require interval refinement. Even for $p = 2$ and $n = 10^5$, refinement is triggered in 1.3% of the cases. (We note in passing that increasing n yields an increase of the number of roots, which are defined on smaller intervals (SI Fig. 8)). In all the cases where refinement was triggered, doubling the precision was sufficient to solve the predicate.

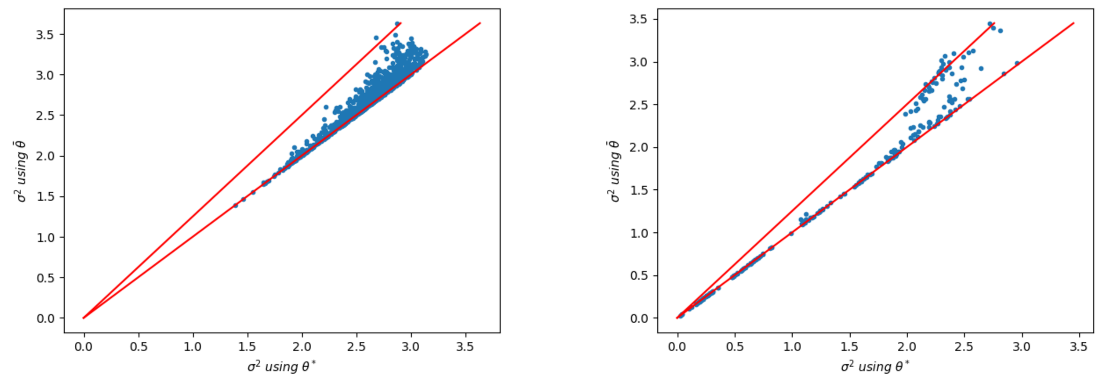
4.3 Fréchet mean

Fréchet mean versus circular mean. A classical way to define the circular mean of a set of angles is the *resultant* or *circular mean*, defined as follows [MJ09]:

$$\overline{\theta} = \text{atan2}(\sum_i \sin\theta_i/n, \sum_i \cos\theta_i/n). \tag{23}$$

The circular mean does not minimize F_p, but minimizes instead [JS01, Section 1.3]:

$$\overline{\theta} = \arg\min \sum_{i=1,\ldots,n} d(\theta_i, \theta), \text{ with } d(\alpha, \beta) = 1 - \cos(\alpha - \beta). \tag{24}$$



Simulated data | Torsion angles from protein structural data

Figure 6: **Variance of angles with respect to the Fréchet mean θ^* and the circular average $\overline{\theta}$. (Left)** Comparison using a simulated set with $n = 30$ angles at random in $[0, 2\pi)$, with 1000 repeats. **(Right)** Comparison for the 243 classes dihedral angles in protein structures–see text. **(Both panels)** In red $y = x$ and $y = 5/4x$.

Given a set of angles, we compare the variance of these angles with respect to the Fréchet mean θ^* and the circular mean $\overline{\theta}$, respectively. Two datasets were used for such experiments: first, randomly generated sets of $n = 30$ angles uniformly at random in $[0, 2\pi)$, with 1000 repeats; second, the aforementioned dihedral angles in protein structures.

For both types of data, the variance obtained for $\overline{\theta}$ is significantly larger than that obtained for θ^*, typically up to 25% (Fig. 6). This shows the interest of using θ^* in data analysis in general, and to center angles prior to principal components analysis in particular.

Remark 6. *While running this experiment, we also counted the type of intervals providing local minima (SI Fig. 11) and the global minimum (SI Fig. 12). Following Rmk. 2, there are indeed 9 possible interval types for (α_j, α_{j+1}). The most probable outcome using these values is $\theta^* \in (\theta_i, \theta_{i+1})$, which owes to the fact that regions of the circle with high sample point density are likely to minimize F_p.*

4.4 Computation time and complexity

The complexity of the algorithm (Theorem. 1) has three main components: the sorting step, the updates of vectors B and C, and the numerics. We wish in particular to determine whether the $n \log n$ term dominates.

For $p \in \{2, 5, 10, 15\}$, we use sets of $n \in [10^3, 10^5]$ angles generated uniformly at random in $[0, 2\pi)$, and perform 5 repeats for each value of n. For $p = 2$, the number of angles is pushed up to $n = 10^7$, with the same number of repeats. In any case, a linear complexity is practically observed (SI Fig. 9, SI Fig. 10), showing that for the values of n used, the constants associated with the linear time update of the data structures and the numerics take over the $n \log n$ term of the sorting step.

5 Software

5.1 Availability

The source code is available in the package *Frechet mean for* S^1 of the Structural Bioinformatics Library (SBL), a library proposing state-of-the art methods in computational structural biology ([CD17] and `https:`

//sbl.inria.fr/). As noticed above, the main motivation in developing this package is to perform an enhanced statistical study of angular values in molecular structures.

Upon cloning the git with *git clone git://sbl.inria.fr/git/sbl.git*, the package is Core/Frechet_mean_S1. Entries to the user manual and the reference manual are https://sbl.inria.fr/doc/Frechet_mean_S1-user-manual.html and https://sbl.inria.fr/doc/group__Frechet__mean__S1-package.html.

The package actually provides two types of software components.

For end-users. We provide two executables, respectively corresponding to the robust and non-robust implementations. Each takes as input a file of angles, the value of p and as an option a corresponding file of weights. A sorted list of pairs (angular value of local minimum, function value) by increasing value of F_p, is then generated. The Jupyter notebook `Frechet_mean_S1.ipynb`, which uses `SAGE` (https://www.sagemath.org/) to visualize F_p and $F_p^{'}$, is also provided.

For developers. The C++ code of our algorithm is provided in the class `SBL::GT::Frechet_mean_S1`, which is templated by the kernel. Two kernels are provided:

- Non-robust kernel: `SBL::GT::Inexact_predicates_kernel_for_frechet_mean`. A plain floating point(double) number type is used.
- Robust kernel: `SBL::GT::Lazy_exact_predicates_kernel _for_frechet_mean`. See Sec. 3.5.

The constructor of this class is templated by an iterator type corresponding to the angular values container. It also requires the value of p, and the value of τ used in the root finding (Algo. 3). The values computed are stored as data members accessible through the corresponding methods.

5.2 Reproducibility of results

To reproduce the results presented in this paper, see Sec. 7.4.

6 Outlook

The Fréchet mean and its generalization the p-mean are of central importance as zero dimensional statistical summaries of data which do not live in Euclidean spaces. For the particular case of S^1, this paper develops the first robust algorithm computing the p-mean of a finite point set. Our algorithm is effective for large number of angular values and large values of p as well, yet, robustness requires predicates and constructions using interval multiprecision arithmetic. For the particular case of the Fréchet mean ($p = 2$), we show that the circular mean should not be used for a substitute to the circular center of mass, as it results in a significantly larger variance.

We foresee two types of types of developments.

For the particular case of S^1, our algorithm is of interest in the context of generalization of principal components analysis (PCA) on the flat torus – an especially important case to deal with molecular data. We also believe that it will prove of interest to improve stochastic algorithms finding p-means for general distributions on S^1, as these may benefit from estimated delivered by our method on random point sets.

In a more general setting, our strategy may be used both to study the intrinsic difficulty of computing p-means (in terms of lower bounds), and to design effective algorithms. Indeed, as evidenced by the S^1 case, the combinatorial structure defined by the cut-loci of the points determines all key properties. A first case would be that of p-means on the unit sphere, for which there exist efficient algorithms to maintain arrangements of circles.

Acknowledgments. Chee Yap and Sylvain Pion are acknowledged for discussions on irrational number theory and number types, respectively.

References

[AJ91] F. Allen and O. Johnson. Automated conformational analysis from crystallographic data. 4. statistical descriptors for a distribution of torsion angles. *Acta Crystallographica Section B: Structural Science*, 47(1):62–67, 1991.

[AM14] M. Arnaudon and L. Miclo. Means in complete manifolds: uniqueness and approximation. *ESAIM: Probability and Statistics*, 18:185–206, 2014.

[AM16] M. Arnaudon and L. Miclo. A stochastic algorithm finding p-means on the circle. *Bernoulli*, 22(4):2237–2300, 2016.

[CCK+06] E-C. Chang, S.W. Choi, D.Y. Kwon, H. Park, and C. Yap. Shortest path amidst disc obstacles is computable. *International Journal of Computational Geometry & Applications*, 16(05n06):567–590, 2006.

[CD17] F. Cazals and T. Dreyfus. The Structural Bioinformatics Library: modeling in biomolecular science and beyond. *Bioinformatics*, 7(33):1–8, 2017.

[cga] Cgal, Computational Geometry Algorithms Library. http://www.cgal.org.

[Cha13] B. Charlier. Necessary and sufficient condition for the existence of a Fréchet mean on the circle. *ESAIM: Probability and Statistics*, 17:635–649, 2013.

[FHL+07] L. Fousse, G. Hanrot, V. Lefèvre, P. Pélissier, and P. Zimmermann. MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Transactions on Mathematical Software (TOMS)*, 33(2):13, 2007.

[Fré48] M. Fréchet. Les éléments aléatoires de nature quelconque dans un espace distancié. In *Annales de l'institut Henri Poincaré*, volume 10, pages 215–310, 1948.

[GK73] K. Grove and H. Karcher. How to conjugatec 1-close group actions. *Mathematische Zeitschrift*, 132(1):11–20, 1973.

[JS01] S.R. Jammalamadaka and A. SenGupta. *Topics in Circular Statistics*. World Scientific, 2001.

[KLPY99] V. Karamcheti, C. Li, I. Pechtchanski, and C. Yap. A core library for robust numeric and geometric computation. In *Proceedings of the fifteenth annual symposium on Computational geometry*, pages 351–359. ACM, 1999.

[KMP+08] L. Kettner, K. Mehlhorn, S. Pion, S. Schirra, and C. Yap. Classroom examples of robustness problems in geometric computations. *Computational Geometry*, 40(1):61–78, 2008.

[KRS16] A. Kobel, F. Rouillier, and M. Sagraloff. Computing real roots of real polynomials... and now for real! In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, pages 303–310. ACM, 2016.

[LPY05] C. Li, S. Pion, and C. Yap. Recent progress in exact geometric computation. *The Journal of Logic and Algebraic Programming*, 64(1):85–111, 2005.

[MBdD+18] J-M. Muller, N. Brunie, F. de Dinechin, C. Jeannerod, M. Joldes, V. Lefèvre, G. Melquiond, N. Revol, and S. Torres. Handbook of floating-point arithmetic. 2018.

[MJ09] K. Mardia and P. Jupp. *Directional statistics*, volume 494. John Wiley & Sons, 2009.

[MT93] M. MacArthur and J. Thornton. Conformational analysis of protein structures derived from NMR data. *Proteins: Structure, Function, and Bioinformatics*, 17(3):232–251, 1993.

[Pen18] X. Pennec. Barycentric subspace analysis on manifolds. *The Annals of Statistics*, 46(6A):2711–2746, 2018.

[PS85] F. Preparata and M. Shamos. *Computational geometry: an introduction*. Springer Science & Business Media, 1985.

[TWS+10] Daniel Ting, Guoli Wang, Maxim Shapovalov, Rajib Mitra, Michael I Jordan, and Roland L Dunbrack Jr. Neighbor-dependent ramachandran probability distributions of amino acids developed from a hierarchical dirichlet process model. *PLoS Comput Biol*, 6(4):e1000763, 2010.

[YD95] C. Yap and T. Dubé. The exact computation paradigm. In *Computing in Euclidean Geometry*, pages 452–492. World Scientific, 1995.

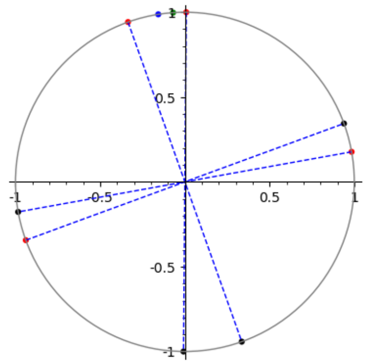
[YYD+10] J. Yu, C. Yap, Z. Du, S. Pion, and Hervé H. Brönnimann. The design of core 2: A library for exact numeric computation in geometry and algebra. In *International Congress on Mathematical Software*, pages 121–141. Springer, 2010.

7 Supporting information

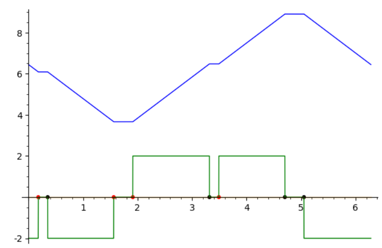
7.1 Illustrations

7.2 Complexity

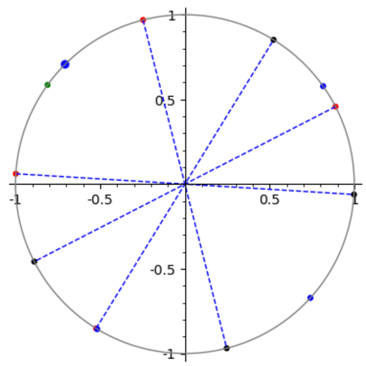
7.3 Intervals determining local and global minima



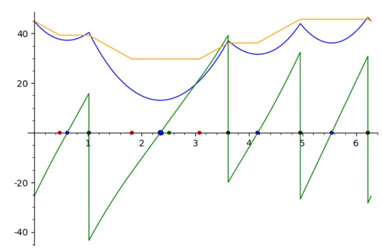
$n = 4, p = 1$



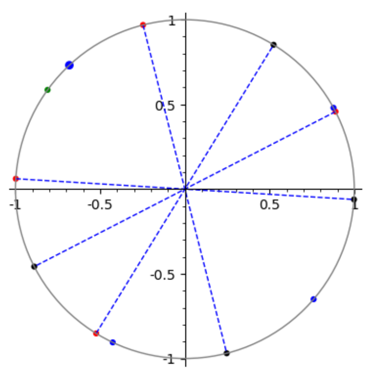
$n = 4, p = 1$



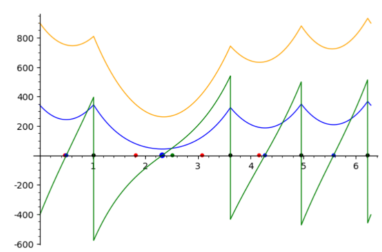
$n = 4, p = 3$



$n = 4, p = 3$



$n = 4, p = 5$



$n = 4, p = 5$

Figure 7: **p-means, illustrations for $p = 1, p = 3, p = 5$ (Top, middle, bottom) rows.** For each row: **(Left panel)** red bullets: points; black bullets: antipodal points; green bullet: mean from resultant i.e. circular mean; blue bullet: p-mean **(Right panel)** blue: function F_p – Eq. 3; green: derivative $F_p^{'}$; orange: second derivative $F_p^{''}$. red bullets: data points; black bullets: antipodal points; blue bullets: local minima of F_p; large blue bullet: p-mean θ^*; green bullet: circular mean $\overline{\theta}$.

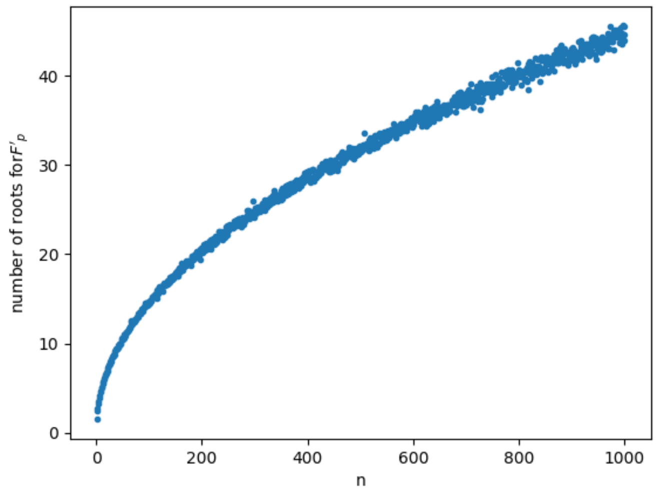


Figure 8: **Fréchet mean: number of roots of $F_p^{'}$ as a function of n for $p = 2$.** Done on point sets generated using uniform distributions in $[0, 2\pi)$

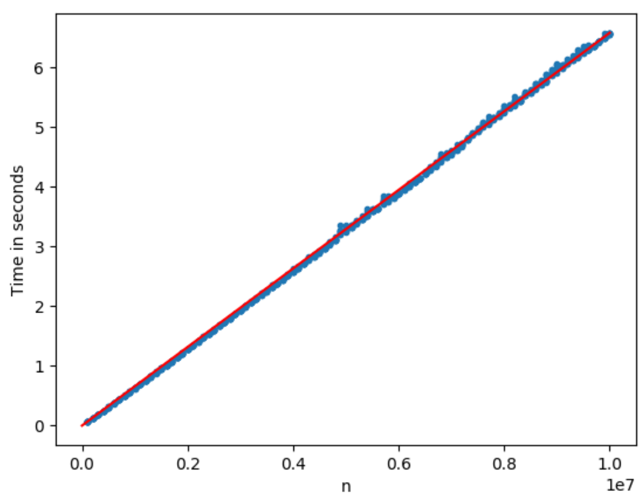


Figure 9: **Fréchet mean: computation time depending as a function of n for $p = 2$.** The red line segments joins $0, 0$ to the average time of the largest point sets.

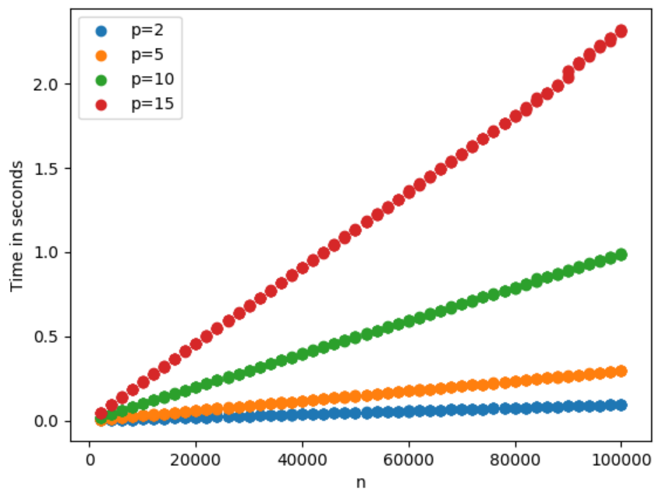


Figure 10: **Fréchet mean: computation time depending as a function of n and p.**

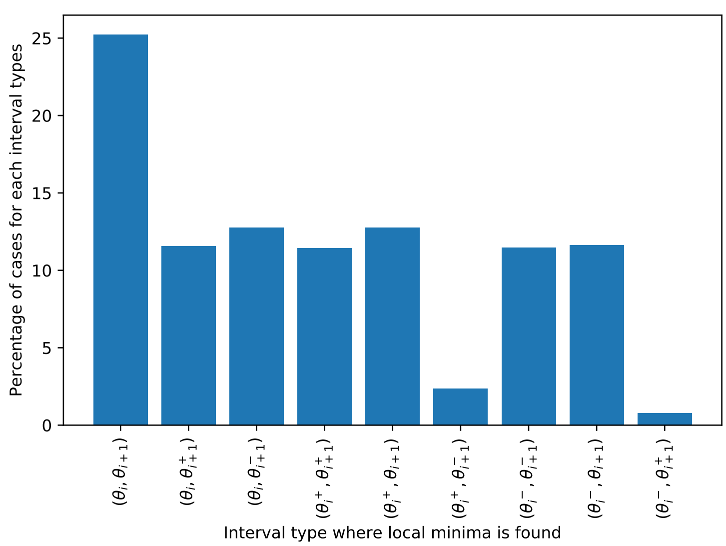


Figure 11: **Type of intervals (α_j, α_{j+1}) yielding local minima.** See Rmk. 2 for the definition of interval types. Statistics for $n = 30$ angles, over 10^6 repeats.

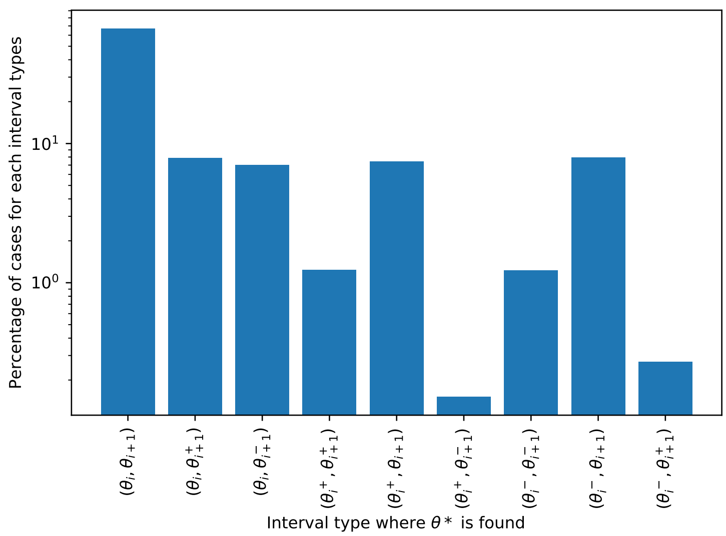


Figure 12: **Type of intervals** (α_j, α_{j+1}) **yielding global minima.** See Rmk. 2 for the definition of interval types. Statistics for $n = 30$ angles, over 10^6 repeats.

7.4 Reproducing the results

Getting the SBL and the package. In the following, we assume that the git of the SBL has been cloned:

```
git clone git://sbl.inria.fr/git/sbl.git
```

On a linux-like environment, we assume the following environment variables:

- $SBL_DIR: the directory containing the sources of the SBL, obtained with git clone as indicated above.
- $SBL_DIR_INSTALL: the directory into which the executables (binaries) are installed. This directory is expected to be in one's PATH.

Programs. cpp code for the executables are provided in the directory `Core/Frechet_mean_S1/src/Frechet_mean_S1/`.
To compile and install these executables, proceed as follows:
• To compile the executables:

```
cd $SBL_DIR/Core/Frechet_mean_S1/src/Frechet_mean_S1/
mkdir build
cd build
cmake .. -DSBL_APPLICATIONS=ON -DCMAKE_INSTALL_PREFIX=$SBL_DIR_INSTALL
make
make install
```

Reproducing the results presented in the paper – running tests. The code to run the tests presented in this paper are provided in the directory `$SBL_DIR/Core/Core/Frechet_mean_S1/examples/`. The program `Frechet_mean_S1-test.cpp` provides options to reproduce all tests presented in the paper:

- –v: tests on the variance
- –i: tests interval type for local minima and the global minimum
- –r: statistics on refinements for numerics
- –c: combinatorial complexity of the algorithm.

```
// Compiling  the test code
cd $SBL_DIR/Core/Frechet_mean_S1/examples/Frechet_mean_S1/
mkdir build
cd build
cmake ..
make

// Running the tests
Frechet_mean_S1-test.exe  -o ../demos/data  -v -i -r -c

// Producing the figures once the tests have been run:
cd $SBL_DIR/Core/Frechet_mean_S1/examples/Frechet_mean_S1

python3 Frechet_mean_S1-test.py  -d ../demos/data
```

Jupyter notebook. Finally, one can also execute the notebook `Frechet_mean_S1.ipynb`, which uses internally `SAGE` (`https://www.sagemath.org/`) to perform the visualization

```
cd $SBL_DIR/Core/Frechet_mean_S1/demos
sage -n jupyter
```

Contents